

DATA MINING

**TOPIC : MARKET BASKET ANALYSIS**

NAME : NATARAJ.C

REGISTER NO : 121012012756

DEGREE : B.TECH

BRANCH : CSE – III YEAR

SUBJECT : DATA MINING

SUB.CODE : XCSE63

XCSE63:DATA MINING

MARKET BASKET ANALYSIS

Name : Nataraj.C

Reg.No :121012012756

Branch : B.Tech-CSE(III Year)

Introduction :

Market Basket Analysis (MBA) is a data mining technique used to uncover associations between items purchased together. It is commonly applied in retail and e-commerce industries to understand customer purchasing behavior and to optimize product placement, promotions, and cross-selling strategies.

The algorithm for Market Basket Analysis is the Apriori algorithm.

1. **Frequent Itemset Generation**: The Apriori algorithm starts by finding frequent itemsets, which are sets of items that frequently occur together in transactions. It begins by identifying individual items' frequencies and gradually extends to larger itemsets by combining frequent itemsets to form candidate itemsets.
2. **Support Counting**: The support of an itemset is the proportion of transactions in which the itemset appears. A minimum support threshold is set, and any itemsets that have support greater than or equal to this threshold are considered frequent itemsets.
3. **Association Rule Generation**: Once frequent itemsets are identified, association rules are generated from these itemsets. An association rule consists of two parts: an antecedent (or left-hand side) and a consequent (or right-hand side). The rule indicates that if the antecedent is present in a transaction, then the consequent is likely to be present as well.
4. **Confidence Calculation**: The confidence of an association rule measures the likelihood of the consequent appearing in a transaction given that the antecedent is present. It is calculated as the support of the combined itemset divided by the support of the antecedent.
5. **Rule Pruning**: Association rules are pruned based on a minimum confidence threshold. Only rules that meet or exceed this threshold are considered strong association rules and are retained for further analysis or application.

By applying the Apriori algorithm, Market Basket Analysis helps businesses identify patterns in customer purchasing behavior and uncover valuable insights that can be used to improve marketing strategies, optimize product assortments, and enhance the overall shopping experience.

#Problem Statement :

Transactions made at the roadside vegetable stand is provided in Table 1. Perform the Market Basket Analysis using Apriori algorithm.

print(data)

#Import necessary libraries

import pandas as pd

from mlxtend.frequent\_patterns import apriori, association\_rules

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

#Read the input csv file

data = pd.read\_csv('Table 1.csv')

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

#Print data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Transaction Asparagus | Beans | Broccoli | Corn | Green | Peppers |
| Squash \ |  |  |  |  |  |
| 0 1 0 | 0 | 1 | 1 | 1 | 0 |
| 0 |  |  |  |  |  |
| 1 2 1 | 0 | 0 | 1 | 0 | 1 |
| 0 |  |  |  |  |  |
| 2 3 0 | 1 | 0 | 1 | 0 | 1 |
| 1 |  |  |  |  |  |
| 3 4 0 | 1 | 0 | 1 | 1 | 0 |
| 1 |  |  |  |  |  |
| 4 5 1 | 1 | 1 | 0 | 0 | 0 |
| 0 |  |  |  |  |  |
| 5 6 1 | 1 | 0 | 0 | 0 | 1 |
| 1 |  |  |  |  |  |
| 6 7 0 | 0 | 0 | 1 | 0 | 0 |
| 1 |  |  |  |  |  |
| 7 8 0 | 0 | 1 | 0 | 1 | 0 |
| 1 |  |  |  |  |  |
| 8 9 1  0 | 1 | 0 | 0 | 0 | 1 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
| 9 |  | 10 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 |  |  |  |  |  |  |  |  |
| 10 |  | 11 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 |  |  |  |  |  |  |  |  |
| 11 |  | 12 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 |  |  |  |  |  |  |  |  |
| 12 |  | 13 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 |  |  |  |  |  |  |  |  |
| 13 |  | 14 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 |  |  | | | | | | |
|  |  |
|  | Tomatoes |
| 0 | NaN |
| 1 | NaN |
| 2 | NaN |
| 3 | NaN |
| 4 | NaN |
| 5 | NaN |
| 6 | NaN |
| 7 | NaN |
| 8 | NaN |
| 9 | NaN |
| 10 | NaN |
| 11 | NaN |
| 12 | NaN |
|  | 13 | NaN |

# Drop unnecessary columns

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

Assuming 'Transaction' is the identifier column

data.drop('Transaction', axis=1, inplace=True)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

print(data)

Asparagus Beans Broccoli Corn Green Peppers Squash Tomatoes 0 0 0 1 1 1 0 0 NaN

1 1 0 0 1 0 1 0 NaN

2 0 1 0 1 0 1 1 NaN

3 0 1 0 1 1 0 1 NaN

4 1 1 1 0 0 0 0 NaN

5 1 1 0 0 0 1 1 NaN

6 0 0 0 1 0 0 1 NaN

7 0 0 1 0 1 0 1 NaN

8 1 1 0 0 0 1 0 NaN

9 0 1 0 1 0 0 0 NaN

10 0 1 1 0 1 1 0 NaN

11 1 1 0 0 0 1 0 NaN

12 1 1 0 1 0 1 0 NaN

13 0 1 1 1 1 0 1 NaN

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code) data.drop('Tomatoes', axis=1, inplace=True)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code) data

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

{"summary":"{\n \"name\": \"data\",\n \"rows\": 14,\n \"fields\": [\n {\n \"column\": \"Asparagus\",\n \"properties\": {\n

\"dtype\": \"number\",\n \"std\": 0,\n \"min\": 0,\n

\"max\": 1,\n \"num\_unique\_values\": 2,\n \"samples\": [\n 1,\n 0\n ],\n \"semantic\_type\":

\"\",\n \"description\": \"\"\n }\n },\n {\n

\"column\": \"Beans\",\n \"properties\": {\n \"dtype\":

\"number\",\n \"std\": 0,\n \"min\": 0,\n

\"max\": 1,\n \"num\_unique\_values\": 2,\n \"samples\": [\n 1,\n 0\n ],\n \"semantic\_type\":

\"\",\n \"description\": \"\"\n }\n },\n {\n

\"column\": \"Broccoli\",\n \"properties\": {\n \"dtype\":

\"number\",\n \"std\": 0,\n \"min\": 0,\n

\"max\": 1,\n \"num\_unique\_values\": 2,\n \"samples\": [\n 0,\n 1\n ],\n \"semantic\_type\":

\"\",\n \"description\": \"\"\n }\n },\n {\n

\"column\": \"Corn\",\n \"properties\": {\n \"dtype\":

\"number\",\n \"std\": 0,\n \"min\": 0,\n

\"max\": 1,\n \"num\_unique\_values\": 2,\n \"samples\": [\n 0,\n 1\n ],\n \"semantic\_type\":

\"\",\n \"description\": \"\"\n }\n },\n {\n

\"column\": \"Green\",\n \"properties\": {\n \"dtype\":

\"number\",\n \"std\": 0,\n \"min\": 0,\n

\"max\": 1,\n \"num\_unique\_values\": 2,\n \"samples\": [\n 0,\n 1\n ],\n \"semantic\_type\":

\"\",\n \"description\": \"\"\n }\n },\n {\n

\"column\": \"Peppers\",\n \"properties\": {\n \"dtype\":

\"number\",\n \"std\": 0,\n \"min\": 0,\n

\"max\": 1,\n \"num\_unique\_values\": 2,\n \"samples\": [\n 1,\n 0\n ],\n \"semantic\_type\":

\"\",\n \"description\": \"\"\n }\n },\n {\n

\"column\": \"Squash\",\n \"properties\": {\n \"dtype\":

\"number\",\n \"std\": 0,\n \"min\": 0,\n

\"max\": 1,\n \"num\_unique\_values\": 2,\n \"samples\": [\n 1,\n 0\n ],\n \"semantic\_type\":

\"\",\n \"description\": \"\"\n }\n }\n ]\ n}","type":"dataframe","variable\_name":"data"}

# Explore the data

print(data.head())

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

Asparagus Beans Broccoli Corn Green Peppers Squash 0 0 0 1 1 1 0 0

1 1 0 0 1 0 1 0

2 0 1 0 1 0 1 1

3 0 1 0 1 1 0 1

4 1 1 1 0 0 0 0

#Get the frequencies of itemsets

itemset\_frequencies = data.apply(pd.Series.value\_counts)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

#Display itemset frequencies

print(itemset\_frequencies)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

Asparagus Beans Broccoli Corn Green Peppers Squash 0 8 4 9 6 9 7 8

1 6 10 5 8 5 7 6

#Apply the support count greater than 4

frequent\_items = itemset\_frequencies[itemset\_frequencies > 4].dropna(axis=1, how='all')

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

#Generate frequent itemsets using Apriori algorithm

frequent\_itemsets = apriori(df, min\_support=0.05, use\_colnames=True)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

/usr/local/lib/python3.10/dist-packages/mlxtend/frequent\_patterns/ fpcommon.py:110: DeprecationWarning: DataFrames with non-bool types

result in worse computationalperformance and their support might be discontinued in the future.Please use a DataFrame with bool type

warnings.warn(

#Apply Confidence of greater than 85%

rules = association\_rules(frequent\_itemsets, metric="confidence", min\_threshold=0.85)

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

# Derive the strong association rules

print(rules)

antecedents consequents antecedent support \

|  |  |  |
| --- | --- | --- |
| 0 | (Peppers) | (Beans) |
| 0.500000 |  |  |
| 1 | (Broccoli, Asparagus) | (Beans) |
| 0.071429 |  |  |
| 2 | (Asparagus, Squash) | (Beans) |
| 0.071429 |  |  |
| 3 | (Corn, Asparagus) | (Peppers) |
| 0.142857 |  |  |
| 4 | (Asparagus, Squash) | (Peppers) |
| 0.071429 |  |  |
| 5 | (Peppers, Broccoli) | (Beans) |
| 0.071429 |  |  |
| 6 | (Green, Peppers) | (Beans) |
| 0.071429 |  |  |
| 7 | (Peppers, Squash) | (Beans) |
| 0.142857 |  |  |
| 8 | (Broccoli, Corn) | (Green) |
| 0.142857 |  |  |
| 9 | (Green, Peppers) | (Broccoli) |
| 0.071429 |  |  |
| 10 | (Peppers, Broccoli) | (Green) |
| 0.071429 |  |  |
| 11 | (Broccoli, Squash) | (Green) |
| 0.142857 |  |  |
| 12 | (Beans, Corn, Asparagus) | (Peppers) |
| 0.071429 |  |  |

1. (Asparagus, Peppers, Squash) (Beans) 0.071429
2. (Asparagus, Beans, Squash) (Peppers) 0.071429
3. (Asparagus, Squash) (Peppers, Beans) 0.071429
4. (Broccoli, Beans, Corn) (Green) 0.071429
5. (Broccoli, Beans, Corn) (Squash) 0.071429
6. (Broccoli, Beans, Squash) (Corn) 0.071429
7. (Broccoli, Corn, Squash) (Beans) 0.071429
8. (Green, Peppers, Broccoli) (Beans) 0.071429
9. (Green, Peppers, Beans) (Broccoli) 0.071429
10. (Peppers, Broccoli, Beans) (Green) 0.071429
11. (Green, Peppers) (Broccoli, Beans) 0.071429
12. (Peppers, Broccoli) (Green, Beans) 0.071429
13. (Broccoli, Beans, Squash) (Green) 0.071429
14. (Green, Beans, Corn) (Squash) 0.142857
15. (Green, Beans, Squash) (Corn) 0.142857
16. (Green, Corn, Squash) (Beans) 0.142857
17. (Peppers, Corn, Squash) (Beans) 0.071429
18. (Broccoli, Corn, Squash) (Green) 0.071429
19. (Green, Broccoli, Corn, Squash) (Beans) 0.071429
20. (Broccoli, Beans, Corn, Squash) (Green) 0.071429
21. (Green, Broccoli, Beans, Corn) (Squash) 0.071429
22. (Green, Broccoli, Beans, Squash) (Corn) 0.071429
23. (Broccoli, Corn, Squash) (Green, Beans) 0.071429
24. (Broccoli, Beans, Corn) (Green, Squash) 0.071429
25. (Broccoli, Beans, Squash) (Green, Corn)

0.071429

consequent support support confidence lift leverage conviction \

0 0.714286 0.428571 0.857143 1.200000 0.071429

2.0

1 0.714286 0.071429 1.000000 1.400000 0.020408

inf

2 0.714286 0.071429 1.000000 1.400000 0.020408

inf

3 0.500000 0.142857 1.000000 2.000000 0.071429

inf

4 0.500000 0.071429 1.000000 2.000000 0.035714

inf

5 0.714286 0.071429 1.000000 1.400000 0.020408

inf

6 0.714286 0.071429 1.000000 1.400000 0.020408

inf

7 0.714286 0.142857 1.000000 1.400000 0.040816

inf

8 0.357143 0.142857 1.000000 2.800000 0.091837

inf

9 0.357143 0.071429 1.000000 2.800000 0.045918

inf

10 0.357143 0.071429 1.000000 2.800000 0.045918

inf

11 0.357143 0.142857 1.000000 2.800000 0.091837

inf

12 0.500000 0.071429 1.000000 2.000000 0.035714

inf

13 0.714286 0.071429 1.000000 1.400000 0.020408

inf

14 0.500000 0.071429 1.000000 2.000000 0.035714

inf

15 0.428571 0.071429 1.000000 2.333333 0.040816

inf

16 0.357143 0.071429 1.000000 2.800000 0.045918

inf

17 0.428571 0.071429 1.000000 2.333333 0.040816

inf

18 0.571429 0.071429 1.000000 1.750000 0.030612

inf

19 0.714286 0.071429 1.000000 1.400000 0.020408

inf

20 0.714286 0.071429 1.000000 1.400000 0.020408

inf

21 0.357143 0.071429 1.000000 2.800000 0.045918

inf

22 0.357143 0.071429 1.000000 2.800000 0.045918

inf

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 23 | 0.214286 | | 0.071429 | 1.000000 | 4.666667 | 0.056122 |
| inf |  | | | | | |
| 24 | 0.214286 | | 0.071429 | 1.000000 | 4.666667 | 0.056122 |
| inf |  | | | | | |
| 25 | 0.357143 | | 0.071429 | 1.000000 | 2.800000 | 0.045918 |
| inf |  | | | | | |
| 26 | 0.428571 | | 0.142857 | 1.000000 | 2.333333 | 0.081633 |
| inf |  | | | | | |
| 27 | 0.571429 | | 0.142857 | 1.000000 | 1.750000 | 0.061224 |
| inf |  | | | | | |
| 28 | 0.714286 | | 0.142857 | 1.000000 | 1.400000 | 0.040816 |
| inf |  | | | | | |
| 29 | 0.714286 | | 0.071429 | 1.000000 | 1.400000 | 0.020408 |
| inf |  | | | | | |
| 30 | 0.357143 | | 0.071429 | 1.000000 | 2.800000 | 0.045918 |
| inf |  | | | | | |
| 31 | 0.714286 | | 0.071429 | 1.000000 | 1.400000 | 0.020408 |
| inf |  | | | | | |
| 32 | 0.357143 | | 0.071429 | 1.000000 | 2.800000 | 0.045918 |
| inf |  | | | | | |
| 33 | 0.428571 | | 0.071429 | 1.000000 | 2.333333 | 0.040816 |
| inf |  | | | | | |
| 34 | 0.571429 | | 0.071429 | 1.000000 | 1.750000 | 0.030612 |
| inf |  | | | | | |
| 35 | 0.214286 | | 0.071429 | 1.000000 | 4.666667 | 0.056122 |
| inf |  | | | | | |
| 36 | 0.214286 | | 0.071429 | 1.000000 | 4.666667 | 0.056122 |
| inf |  | | | | | |
| 37 | 0.214286 | | 0.071429 | 1.000000 | 4.666667 | 0.056122 |
| inf |  |
|  |

#Write the Learning Summary

learning\_summary = """

Market Basket Analysis using Apriori Algorithm:

* Explored transaction data to understand item frequencies.
* Applied Apriori algorithm to generate frequent itemsets with a minimum support of 5%.
* Derived association rules with a minimum confidence of 85%. """

print(learning\_summary)

Market Basket Analysis using Apriori Algorithm:

* Explored transaction data to understand item frequencies.
* Applied Apriori algorithm to generate frequent itemsets with a minimum support of 5%.
* Derived association rules with a minimum confidence of 85%.

#Calculate accuracy as confidence

rules['accuracy'] = rules['confidence']

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code) rules

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

{"repr\_error":"Out of range float values are not JSON compliant: inf","type":"dataframe","variable\_name":"rules"}

#Convert to dataframe

from matplotlib import pyplot as plt

import seaborn as sns

def \_plot\_series(series, series\_name, series\_index=0):

palette = list(sns.palettes.mpl\_palette('Dark2')) counted = (series['accuracy']

.value\_counts()

.reset\_index(name='counts')

.rename({'index': 'accuracy'}, axis=1)

.sort\_values('accuracy', ascending=True))

xs = counted['accuracy']

ys = counted['counts']

plt.plot(xs, ys, label=series\_name, color=palette[series\_index % len(palette)])

accuracy\_data = pd.DataFrame(rules[['antecedents', 'consequents', 'accuracy']])

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code) accuracy\_data

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

fig, ax = plt.subplots(figsize=(10, 5.2), layout='constrained') df\_sorted = accuracy\_data.sort\_values('accuracy', ascending=True)

\_plot\_series(df\_sorted, '') sns.despine(fig=fig, ax=ax) plt.xlabel('accuracy')

\_ = plt.ylabel('count()')

<google.colab.\_quickchart\_helpers.SectionTitle at 0x7e3e5e1c3e80>

from matplotlib import pyplot as plt accuracy\_data['accuracy'].plot(kind='line', figsize=(8, 4), title='accuracy')

plt.gca().spines[['top', 'right']].set\_visible(False)

#Visualize highly associated values

import matplotlib.pyplot as plt

*# Get the top 10 most accurate rules*

top\_10\_accurate\_rules = accuracy\_data.sort\_values(by='accuracy', ascending=False).head(10)

*# Plot the bar chart*

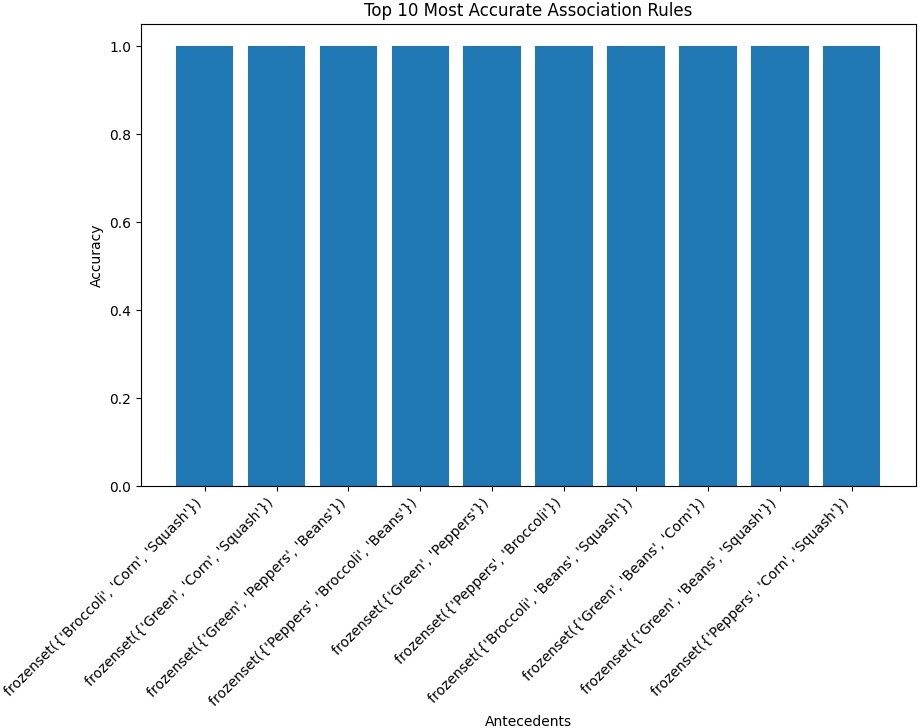
plt.figure(figsize=(10, 6)) plt.bar(top\_10\_accurate\_rules['antecedents'], top\_10\_accurate\_rules['accuracy']) plt.xticks(rotation=45, ha='right') plt.xlabel('Antecedents') plt.ylabel('Accuracy')

plt.title('Top 10 Most Accurate Association Rules') plt.show()

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)



accuracy\_data['antecedents'] = accuracy\_data['antecedents'].apply(lambda x: str(x)) accuracy\_data['consequents'] = accuracy\_data['consequents'].apply(lambda x: str(x))

plt.figure(figsize=(10, 6)) plt.scatter(accuracy\_data['antecedents'], accuracy\_data['consequents'], s=accuracy\_data['accuracy']\*1000, alpha=0.5)

plt.xlabel('Antecedents') plt.ylabel('Consequents') plt.title('Highly Associated Values') plt.xticks(rotation=90) plt.yticks(rotation=90) plt.grid(True)

plt.show()

/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically in the future. Please pass the result to

`transformed\_cell` argument and any exception that happen during thetransform in `preprocessing\_exc\_tuple` in IPython 7.17 and above.

and should\_run\_async(code)

